REMARKS

Applicant respectfully requests reconsideration of the present application in view of the reasons that follow.

Claims 21, 27, 30, 32, 38, and 41 have been amended, and claims 29 and 40 have been canceled. Accordingly, claims 21-28, 30-39, and 41-42 remain pending in this application. A detailed listing of all claims that are, or were, in the application, irrespective of whether the claim(s) remain under examination in the application, is presented, with an appropriate defined status identifier.

Claims 21 and 32 have been amended to incorporate the recitations of canceled claims 29 and 40 respectively. Claims 27 and 38 have been amended to be in independent format.

In the Office Action, claims 21-24 and 32-35 were rejected under 35 U.S.C. § 102(e) as being anticipated by Srivastava (U.S. Patent No. 5,999,737). Claims 25 and 36 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Srivastava in view of Mulchandani et al. (U.S. Patent No. 6,112,025). Claims 26 and 37 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Srivastava in view of Bacon et al. (U.S. Patent No. 6,041,179. This rejections are moot in view of the amendments to claims 21 and 32.

Claims 27-31 and 38-42 were rejected under 35 U.S.C. § 103(a) as being unpatentable over Srivastava (U.S. Patent No. 5,999,737) in view of Click, Jr. et al. (U.S. Patent No. 6,408,433). Claim 21, as amended, recites that a method for removing dead code in code fragments of a program comprises identifying each instruction assigning a register that is possibly live for each exit in a first code fragment, storing information corresponding to each instruction identified for the corresponding exit in an epilog associated with each exit of the first code fragment, and identifying each register that is assigned before being read in a second code fragment having a first entry. The method further comprises, at a time when linking a first exit from the first code fragment to the first entry in the second code fragment, comparing the registers in the instructions identified as being possibly live in the first code fragment with the identified registers in the second code fragment, and eliminating an instruction in the first code fragment based on the comparison.

With respect to claim 29 (applicable to claim 21 as amended), the Examiner asserted that although Srivastava discloses storing information corresponding to the instructions so as to remove dead code, it fails to disclose or suggest storing information corresponding to each instruction identified for the corresponding exit in an epilog associated with each exit of the first code fragment. The Examiner further asserted that Click discloses using a register allocator to build prolog and epilog code so as to allow for greater flexibility and faster execution. Finally, the Examiner asserted it would have been obvious to store the information of Srivastava in an epilog as taught by Click to allow greater flexibility and faster execution.

Applicant respectfully disagrees with the Examiner's assertions. First of all, although Srivastava discloses performing a liveliness analysis, it in no way discloses or suggest storing the information derived from the analysis. In column 10, lines 1-14, Srivastava merely discloses that the liveliness information allows for removing dead code, but does not disclose or suggest that the information is stored.

Click does indeed disclose that the use of a register allocator to generate calling convention code (such as prolog and epilog code as noted in column 6, lines 44-45) increases the flexibility of overall code and enables the code to execute more quickly. However, it is entirely unclear how storing the liveliness information of Srivastava in an epilog as taught by Click would provide the benefit of increased flexibility and more quickly executed code. In other words, not only does Srivastava fail to disclose or suggest storing the liveliness information, but there is no disclosure or suggestion in Click as to how storing that information in an epilog would provide the benefits of increased flexibility and more quickly executed code. While Click discloses motivation for including a register allocator that generates prologs and epilogs, neither Srivastava nor Click suggests storing liveliness information in the prologs or epilogs or provides any reason or motivation for doing so. Accordingly, even if combinable, the combination of Srivastava and Click fails to disclose or suggest storing information corresponding to each instruction identified for the corresponding exit in an epilog associated with each exit of the first code fragment as recited in claim 21.

Claim 21 is therefore patentably distinguishable from the combination of Srivastava and Click.

Claims 22-26 and 30-31 are patentably distinguishable from the combination of Srivastava and Click by virtue of their dependence from claim 21, as well as their additional recitations. Claims 32-37 and 41-42 are patentably distinguishable from the combination of Srivastava and Click for reasons analogous to claim 21.

Claim 27, as amended, recites that method for removing dead code in code fragments of a program comprises identifying each instruction assigning a register that is possibly live for a first exit in a first code fragment, and generating a first register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is assigned in an instruction identified in the first code fragment. The method further comprises identifying each register that is assigned before being read in a second code fragment having a first entry, and generating a second register mask, the second register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is one of the identified registers in the second code fragment.

With respect to claim 27, the Examiner asserted that although Srivastava discloses determining live ranges of variables so to remove dead code but admitted that Srivastava fails to disclose or suggest generating the first and second register masks. In addition, the Examiner asserted that Click discloses register masks where bits are set if the register at that position is valid and that it would have been obvious to supplement the method of Srivastava with register masks as taught by Click to provide an additional mean by which to determine live ranges.

Applicant again respectfully disagrees with the Examiner's assertions. Although Srivastava discloses generating liveliness information, there is nothing in Srivastava that discloses or suggests anything about live ranges of variables. In fact, the term "live range" does not appear anywhere in the specification of Srivastava. The reference cited by the examiner (column 10, lines 1-14) merely discloses that the liveliness information (which

identifies variables as being live at the start and at the end of a block) allows for removing dead code, not to determine live ranges.

Click does indeed disclose the use of register masks to assist in determining live ranges. In particular, Click discloses that a register mask 452 includes multiple bits 460, each bit 460 set to indicate whether a particular register is valid with respect to the variable with which the register mask 452 is associated (col. 7, lines 50-53). When a bit 460 is set to a value of "1," the register associated with the bit is valid, and when set to a value of "0,", the register associated with the bit is invalid (col. 7, lines 55-60). In one embodiment, at most one bit is set, and in another embodiment, two bits may be set, as for example when bits 460 represent long integers (col. 7, lines 60-64). Thus, Click discloses that each register mask is associated with a variable, and at most one or two bits can be set.

In contrast to claim 27, however, Click does not disclose or suggest generating a first register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is assigned in an instruction identified in the first code fragment. Rather, Click discloses setting a bit if the register is valid, not if the respective register is assigned and identified as being possible live in a first code fragment as recited in claim 27. There is nothing in Click that discloses or suggest that the validity of a register has anything to do with whether the respective register is assigned and identified as being possible live in a first code fragment. Moreover, the fact that no more than one or two bits can be set in Click proves that they are unrelated as more than two registers can be set in the register masks of claim 27.

Similarly, Click fails to disclose or suggest generating a second register mask, the second register mask having a plurality of positions, each position corresponding to a respective register, wherein a bit at a position is set if the respective register is identifying as being assigned before being read in a second code fragment as recited in claim 27. There is nothing in Click that discloses or suggest that the validity of a register has anything to do with whether the respective register is assigned before being read in a second code fragment.

Accordingly, even if combinable, claim 27 is patentably distinguishable from the combination of Srivastava and Click. Claim 28 is patentably distinguishable from the combination of Srivastava and Click by virtue of its dependence from claim 27, as well as its additional recitations. Claims 38-39 are patentably distinguishable from the combination of Srivastava and Click for reasons analogous to claim 27.

Applicants believe that the present application is now in condition for allowance. Favorable reconsideration of the application as amended is respectfully requested.

The Examiner is invited to contact the undersigned by telephone if it is felt that a telephone interview would advance the prosecution of the present application.

Respectfully submitted,

Date 4/5/05 By _____

William T. Ellis
Attorney for Applicants
Registration No. 26,874
Telephone: (202) 672-5485
Facsimile: (202) 672-5399